ENCRYPTION PROCESSING FOR STREAMING MEDIA

FIELD

This invention relates to encryption techniques for streaming media.

PRIOR ART

Digital content such as video, audio and multimedia digital files are usually quite large and there may be a long transmission delay before a recipient is able to see or hear the content of such files when transmitted to him, particularly over the internet where bandwidth at some point in the transmission channel may be limited. The technique of streaming media files addresses this problem and allows the recipient to commence viewing video or listening to audio long before the media file has been completely received by him. A video may be displayed and audio played as soon as there is sufficient data in the buffer of the recipient station.

As with other digital files there are a number of applications where it is highly desirable for a streaming media file to be encrypted. Usually the content will be subject to copyright and encryption will be desired to supplement the legal protection. The encryption of streaming media poses additional problems to those encountered when a complete file is encrypted, transmitted and decrypted before use by the recipient. In practice most digital data transmission over a distance is conducted using packet technology. Therefore the encryption of streaming media requires packet by packet encryption and decryption if the advantages of streaming are to be preserved.

SUMMARY OF INVENTION

It is an object of the present invention to provide fast and secure encryption for streaming media.

Accordingly the invention consists in a method for securely transmitting streamed media consisting of a stream header and a series of data packets between a sender and a recipient comprising the steps of:

generating a random base key,

encrypting said base key to create an open key,

transmitting said open key to said recipient,

encrypting each packet of the data in the stream by:

(a) assigning a tag value to each packet if no tag value already exists,

(b) creating a packet key by computing a secure hash of said base key and the tag value or the assigned tag value of the packet,

(c) encrypting the data in the packet using said packet key, and

(d) adding said tag value to the corresponding encrypted packet data and inserting the packet so processed into the packet stream,

transmitting the encrypted packet stream to said recipient,

at the recipient's station receiving said open key and the encrypted packet stream,

decrypting said open key to derive the base key,

decrypting each received encrypted packet in the stream by:

(a) extracting the tag value from each packet,

(b) recreating said packet key by computing a secure hash of the base key and the packet's tag value,

(c) decrypting the packet data using said packet key,

(d) and storing or outputting the decrypted packet data in a form suitable for playing the streamed media.

The invention consists in the foregoing and also envisages constructions of which the following gives examples.

BRIEF DESCRIPTION OF DRAWINGS

Figure 1 is a block diagram showing the essential encryption and decryption process steps of the present invention.

DESCRIPTION OF THE PREFERRED EMBODIMENT

In this patent specification:

(1) a "stream" is assumed to be composed of a stream header, containing information about the stream itself, followed by a series of packets,

(2) each "packet" is assumed to consist of a packet header plus the actual data payload of the packet,

(3) the generated random key used to create and deconstruct packet keys is called a "base key" but could also be termed a "session key", a "base stream key" or a "closed key",

(4) both the packet header and data payload are assumed to consist of some number of entire bytes (octets), which numbers may vary from packet to packet, and

(5) each packet header is assumed to include at least one item of information that uniquely identifies that packet, called here a "tag". An example could be a time position value indicating the packet's relative time offset from the stream start, or a serial number, or (x, y) co-ordinates pertaining to the intended destination or location of the packet data. The tag information, along with the rest of the packet header, must accompany a packet "in the clear", that is, not encrypted.

Figure 1 shows diagrammatically the steps of the present encryption system as applied by a sender of streaming media and the steps of decryption as applied by a recipient of the streaming media. This may be implemented in various preferred manners as described below.

The encryption process essentially involves encrypting (6) the data in each streaming media packet with a packet key which is generated by computing a secure hash (5) of the packet tag value with a base key. To decrypt the encrypted packet data it is necessary for the recipient to be able to generate the packet key and in order to do this the recipient must be in possession of the base key. The base key generated (1) randomly by the sender is passed to the recipient after being asymmetrically encrypted (3) with the recipient's public key (2). The encrypted base key (open key) is transmitted to the recipient. It is convenient, although not necessary, to pass the open key to the recipient in the stream header as indicated in Figure 1 by the dotted lines.

The packet encryption process commences with the extraction of tag data from incoming streaming media packets (or the creation of such tags if the packets do not contain tag data) and creating a packet key by computing a secure hash of the base key and the tag value. For each streamed media transmission or session, a base or session key (1) is randomly generated and is hashed (5) with the tag value of each streaming media packet to produce a packet key which is used to encrypt (6) (preferably

symmetrically) the data carried in the corresponding packet. Hash functions such as SHA-1 or MD5 are suitable for the hash computation. While in the preferred embodiment packet data is encrypted using a symmetric algorithm, other types of encryption algorithms could be used so long as the required minimum computation time is not exceeded. The encrypted packet data or ciphertext is then combined (7) with the unencrypted or open tag value for that packet. The process described is applied to each data packet in the stream to produce an encrypted packet stream which is transmitted to the recipient.

At the recipient's station the stream header is first received and the open key extracted from it unless the open key has already been received earlier by other means. The open key is then asymmetrically decrypted (12) using the recipient's private key 11. The decrypted open key, that is the base key, is then available to the hash algorithm (14) to allow the packet key to be derived.

The tag values of each stream data packet are extracted (13) and then hashed (14) with the base key to produce the packet key for each packet. The stream packets with tag values removed (stream data) are then symmetrically decrypted (15) using the corresponding packet key. The plaintext stream packets, with or without tag values depending on the transmission protocol being used, are then stored or outputted in a form suitable for use by a streaming media player.

The present encryption processing may insert specific information into designated field(s) within the stream header, and also replaces the data payload of each packet with encrypted data. All of the packets in the stream are encrypted, but only the data payload is encrypted and not the packet header information. This remains unchanged by the encryption processing. The size in bytes of the encrypted data in a packet need not be larger than the size of the original (unencrypted) data payload.

The encryption algorithm used to protect the base key prior to its transfer from sender to recipient preferably is a public-key (asymmetric) algorithm in which the recipient's public key is used by the sender to encrypt the packet key and the corresponding private key is used by the recipient to decrypt it. The private keys used by the public key algorithm may contain protocol or formatting information in addition

to key data bits. The number of key data bits (the key size) is scalable for different applications, based on security requirements, but at least 607 bit keys must be allowable. A specific application may permit only a single key size to be used, or may handle multiple key sizes.

A valid private key preferably consists of any specified (arbitrary or application-defined, or randomly generated) pattern of key data bits, although a limited number of data patterns may be excluded (for example, all zeros or all ones). Public keys should have the same number of key data bits as private keys, and a fast asymmetric algorithm is used to rapidly generate a public key from any valid private key, while at the same time it is computationally infeasible to ascertain the private key from which a particular public key has been generated.

When encrypting a stream using a particular public key, the encryption algorithm (3) may insert data comprising the open key into an appropriate designated place within the stream header as shown by the dotted lines in Figure 1, or alternatively the open key can be transmitted to the recipient at an earlier time as shown by the solid lines between encryption Stage 3 and decryption Stage 12. Upon receipt of the open key, a recipient who possesses the correct private key is able, using the decryption algorithm, to use the open key to establish or calculate information (a "closed key") known only to the sender and to the recipient(s).

The process of encrypting each packet's data depends upon (ie. utilises in an essential way) the closed key and the packet's tag information. It is computationally infeasible, without knowing the private key, to calculate or determine the closed key based upon knowledge of publicly accessible information such as the public key, open key, encrypted stream data, packet tags, etc.

The encryption processing should use true random data to ensure that the closed key created for encrypting any stream varies randomly even if the same stream is encrypted on multiple occasions. The encryption algorithm ensures that even if two packets of data in the original stream contain identical data bits, the encrypted data for the packets is different. The encryption processing also ensures that if a stream is

encrypted on multiple occasions, the encrypted packet data and the open key are different on each occasion.

The decryption processing is such that each intact packet that is received can be correctly decrypted, even if some packets are not received, or if some or all packets are received in a different order from the order in which they were sent.

In use a content owner of the streamed audio visual works will be provided with software which performs the encryption process as and when content is distributed. The software input will take the form of a packetised, compressed and encoded file. For example if the stream was to be played by a recipient using a Real player the file would use Real coding. Standard compression algorithms such as MPEG2 or MPEG4 would be used. In the case of a Real encoded file the tag value, which is an essential part of the present encryption system, would be the Real timeline value. This is a measure of time from when the stream starts.

Users will have received decryption software in advance from the content provider direct, by downloading from a website or bundled with the streaming player software. A typical business model for deployment of the present encryption software would be for content providers to be licensed for value with users being provided with the decryption software component free of charge. Users would pay the content provider for only the content itself. On payment (or equivalent) for the content the user would be provided with the open key corresponding to the requested content.

A system of media key management and distribution suitable for use in conjunction with the present encryption system is disclosed in applicant's New Zealand patent application 509037.